UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/706,216	11/12/2003	Fei Luo	BEAS-01339US3	6317

23910 7590 01/25/2008

FLIESLER MEYER LLP
650 CALIFORNIA STREET
14TH FLOOR
SAN FRANCISCO, CA 94108

EXAMINER	
ZHEN, LI B	

ART UNIT	PAPER NUMBER
2194	

MAIL DATE	DELIVERY MODE
01/25/2008	PAPER

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

Office Action Summary	Application No.	Applicant(s)	
	10/706,216	LUO ET AL.	
	Examiner	Art Unit	
	Li B. Zhen	2194	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *08 November 2007*.

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *10,17 and 19-42* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *10,17 and 19-42* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☒ Information Disclosure Statement(s) (PTO/SB/08) Paper No(s)/Mail Date 11/8/2007
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____.
5) ☐ Notice of Informal Patent Application
6) ☐ Other: _____.

DETAILED ACTION

1. Claims 10, 17 and 19 – 42 are pending in the application.

Response to Arguments

2. Applicant's arguments with respect to the claims have been considered but are moot in view of the new ground(s) of rejection.

Claim Rejections - 35 USC § 103

3. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

4. This application currently names joint inventors. In considering patentability of the claims under 35 U.S.C. 103(a), the examiner presumes that the subject matter of the various claims was commonly owned at the time any inventions covered therein were made absent any evidence to the contrary. Applicant is advised of the obligation under 37 CFR 1.56 to point out the inventor and invention dates of each claim that was not commonly owned at the time a later invention was made in order for the examiner to consider the applicability of 35 U.S.C. 103(c) and potential 35 U.S.C. 102(e), (f) or (g) prior art under 35 U.S.C. 103(a).

5. **Claims 10, 17 and 19 – 42 are rejected under 35 U.S.C. 103(a) as being unpatentable over U.S. Patent No. 6,993,774 to Glass [previously cited] in view of U.S. Patent No. 6,877,163 to Jones et al. [hereinafter Jones].**

6. As to claim 10, Glass as modified teaches the invention substantially as claimed including a method for processing an invocation [col. 11, line 60 – col. 12, line 5] using a dynamically generated wrapper [dynamic generation of remote proxies; col. 6, lines 40 – 55], comprising:

receiving an invocation by a wrapper object [In order to isolate the distributed processing communication requirements from local object 20, a remote proxy object 22 may be created on server system 12 and loaded onto client system 14; col. 5, line 52 – col. 6, line 7], the wrapper object instantiated from a wrapper class [col. 8, lines 47 – 57], the wrapper class extended from a superclass [col. 8, lines 30 – 40], the invocation directed to a wrapped resource adapter [proxy object 22 which contains the interfaces; col. 6, lines 40 – 55] by an application [Local object 20 may request access to subject object 18; col. 5, line 52 – col. 6, line 7];

initiating pre-processing by the wrapper object [Type object 204 forwards the message to the appropriate EJB function object 206 for preliminary processing; col. 15, lines 38 – 56; col. 13, line 57 – col. 14, line 15];

calling the wrapped resource adapter [proxy object 22 which contains the interfaces; col. 6, lines 40 – 55] by the wrapper object [Local object 20 communicates

with remote proxy object 22 which then communicates with subject object 18; col. 5, line 52 – col. 6, line 7];

receiving a result from the wrapped resource adapter [passes a result through server-side ORB 114 across network; col. 13, lines 40 – 58] by the wrapper object [Reference object 158 decodes the result and passes it to remote proxy 154; col. 13, lines 40 – 58];

initiating post-processing by the wrapper object[Set of streamers 180 handles the encoding and transmission...results; col. 14, lines 13 – 31 and col. 15, lines 56 – 67; Client-side ORB 112 locates the appropriate reference object 158 utilizing communication protocol information received with the result message, col. 15, lines 1 – 16]; and

provide the result to the application [Remote proxy 154 then makes the result available to client application 108; col. 13, lines 40 – 58].

Although Glass teaches the invention substantially, Glass does not specifically disclose a wrapper class extended from a superclass which implements a predefined wrapper interface that includes a pre-invocation handler and a post-invocation handler, the pre-processing code includes calling the pre-invocation handler, the pre-invocation handler is configured to execute server-side code, the server-side code includes transaction processing code, post-processing including calling the post-invocation handler, the post-invocation handler is configured to perform post-processing server side tasks and the post-processing server-side tasks include transaction management.

However, Jones teaches a wrapper class extended from a superclass which implements a predefined wrapper interface [a proxy class, dynamically generated at runtime, that implements a list of interfaces specified at runtime; col. 3, lines 5 – 15] that includes a pre-invocation handler and a post-invocation handler [proxy class instance automatically encodes and dispatches a method invocation of a method on an interface implemented by the proxy class instance to an invocation handler object that automatically handles the request and returns the result; col. 3, lines 28 – 43], the pre-processing code includes calling the pre-invocation handler [col. 4, lines 20 – 35], the pre-invocation handler is configured to execute server-side code that includes transaction processing code [col. 4, lines 20 – 35], post-processing including calling the post-invocation handler [col. 3, lines 28 – 43], the post-invocation handler is configured to perform post-processing server side tasks and the post-processing server-side tasks include transaction management [col. 6, lines 42 – 63].

It would have been obvious to a person of ordinary skill in the art at the time the invention was made to modify the invention of Glass to incorporate the features of Jones because this facilitates the processing of method invocations to a single object that uniformly handles method invocations of varying types of multiple interfaces [col. 6, line 62 – col. 7, line 3 of Jones].

7. As to claim 17, Glass as modified teaches a method for dynamically generating a wrapper object [dynamic generation of remote proxies; col. 6, lines 40 – 55 of Glass], comprising:

receiving a resource adapter class [reads the associated class 252 from a class repository, col. 18, lines 56 – 63 of Glass, see Fig. 11, element 252 can be either class or object; Glass also discloses locating the subject object, step 26, Fig. 2, col. 7, lines 19 – 35; Examiner notes that the specification does not specifically define a vendor object, therefore a vendor object is given its plain meaning and is interpreted as object that provides services to other applications. The subject object as disclosed in Glass exists on a server system and provides services to clients, see col. 8, lines 1 – 12. Therefore, the subject object as disclosed in Glass corresponds to the recited vendor object.] at an application server [server systems 12; col. 4, line 62 – col. 5, line 8 of Glass];

performing reflection on the resource adapter class [invokes reflection engine 36 to determine information regarding subject class 19; col. 8, lines 1 – 12 of Glass] to identify interfaces implemented by the resource adapter class [proxy object 22 which contains the interfaces; col. 6, lines 40 – 55 of Glass];

dynamically generating a wrapper class at runtime [generate the byte codes that define the class of subject object 18, col. 6, line 55 – col. 7, line 6 of Glass; remote proxy for the subject object will inherit all of the variables and methods of its ancestors; col. 7, lines 58 – 67 of Glass] that extends from a superclass [col. 8, lines 30 – 40 of Glass], wherein the superclass implements a predefined wrapper interface [a proxy class, dynamically generated at runtime, that implements a list of interfaces specified at runtime; col. 3, lines 5 – 15 of Jones] that includes a pre-invocation handler and a post-invocation handler [proxy class instance automatically encodes and dispatches a

method invocation of a method on an interface implemented by the proxy class instance to an invocation handler object that automatically handles the request and returns the result; col. 3, lines 28 – 43 of Jones], and the wrapper class implements the interfaces identified through reflection [col. 8, lines 11 – 30 of Glass];

instantiating a wrapper object from the wrapper class [class loader 46 takes the generated bytes of remote proxy class 23 stored in memory and loads them into a class structure which then can be instantiated to create remote proxy object 22; col. 10, lines 1 – 10 of Glass]; and

providing the wrapper object [generated interface is associated with subject class 19; col. 8, lines 40 – 48 of Glass] to an application that requires support for the interfaces implemented by the resource adapter class [col. 6, lines 40 – 55 of Glass].

8. As to claim 19, Glass as modified teaches initiating pre-processing by the wrapper object, wherein the pre-processing code includes calling a pre-invocation handler, wherein the pre-invocation handler is configured to execute server-side code [col. 15, lines 38 – 56 of Glass], wherein the server-side code includes transaction processing code [col. 4, lines 20 – 35 of Jones].

9. As to claim 20, Glass as modified teaches initiating post-processing by the wrapper object, wherein post-processing including calling a post-invocation handler, wherein the post-invocation handler is configured to perform post-processing server

side tasks [col. 14, lines 13 – 31 of Glass], wherein the post-processing server-side tasks include transaction management [col. 6, lines 42 – 63 of Jones].

10. As to claim 21, Glass as modified teaches the wrapper object is a proxy generated at runtime [col. 3, lines 5 – 15 of Jones] and acts as a delegate for an underlying vendor object [col. 3, lines 15 – 29 of Jones].

11. As to claim 22, Glass as modified teaches the wrapper object is used to intercept method invocations from an application program to a vendor object [col. 3, lines 5 – 15 of Jones] and provide for execution of server side tasks in a pre-invocation handler and a post-invocation handler [col. 3, lines 28 – 43 of Jones].

12. As to claim 23, Glass as modified teaches the wrapper object is used to intercept a method invocation against the vendor object [invocation handler; col. 5, lines 5 – 23 of Jones].

13. As to claim 24, Glass as modified teaches the wrapper object provides for server side tasks to be performed before sending a wrapped result to the application [col. 5, lines 22 – 39 of Jones].

14. As to claim 25, Glass as modified teaches the wrapper object is dynamically generated at runtime by a wrapper factory on an application server [col. 6, lines 5 – 23 of Jones].

15. As to claim 26, Glass teaches retrieved meta information from performing reflection allows an application server to dynamically generate a wrapper class that perfectly matches the vendor class [col. 8, lines 1 – 12].

16. As to claim 27, Glass as modified teaches a wrapper class includes all public interfaces implemented by a vendor class and required by the application [col. 5, lines 5 – 22 of Jones].

17. As to claim 28, Glass as modified teaches the application can cast the wrapper object to a vendor interface to access vendor extension methods [cast to any of the interfaces * specified when the proxy class was created; col. 8, lines 25 – 35 of Jones].

18. As to claim 29, Glass as modified teaches the application server has code for dynamically generating the wrapper [col. 6, lines 40 – 55 of Glass and col. 3, lines 5 – 15 of Jones].

19. As to claim 30, Glass as modified teaches a wrapper factory uses a static method to dynamically generate a wrapper [col. 7, line 3 – col. 8, line 8 of Jones].

20. As to claim 31, Glass as modified teaches the superclass has a member variable to hold a vendor object, a non-argument constructor to instantiate the wrapper object, and an init method to initialize the wrapper object [creates the proxy class by calling the getProxyClass method of the other class and passing representations of the desired interfaces as arguments to that method; col. 5, lines 5 – 22 of Jones].

21. As to claims 32 – 42, see the rejections to claims 21 – 31 above.

CONTACT INFORMATION

22. Any inquiry concerning this communication or earlier communications from the examiner should be directed to Li B. Zhen whose telephone number is (571) 272-3768. The examiner can normally be reached on Mon - Fri, 8:30am - 5pm.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, William Thomson can be reached on 571-272-3718. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free). If you would like assistance from a USPTO Customer Service Representative or access to the automated information system, call 800-786-9199 (IN USA OR CANADA) or 571-272-1000.

Li B. Zhen
Primary Examiner
Art Unit 2194

lbz